Detection of Duplicated Regions in Tampered Digital Images by Bit-Plane Analysis

Edoardo Ardizzone and Giuseppe Mazzola

Dipartimento di Ingegneria Informatica (DINFO) - Università degli Studi di Palermo
Viale delle Scienze, building 6, 90128, Palermo, Italy
ardizzon@unipa.it, mazzola@dinfo.unipa.it

Abstract. In this paper we present a new method for searching duplicated areas in a digital image. The goal is to detect if an image has been tampered by a copy-move process. Our method works within a convenient domain. The image to be analyzed is decomposed in its bit-plane representation. Then, for each bit-plane, block of bits are encoded with an ASCII code, and a sequence of strings is analyzed rather than the original bit-plane. The sequence is lexicographically sorted and similar groups of bits are extracted as candidate areas, and passed to the following plane to be processed. Output of the last planes indicates if, and where, the image is altered.

Keywords: Image Forensics, Image Analysis, Bit-Plane Decomposition, Duplication Detection, Image Forgeries.

1 Introduction and Previous Works

A picture is worth a thousand words. But sometimes it does not tell the truth. Nowadays new digital techniques and tools (i.e. Adobe Photoshop) make it relatively easy to alter the content of a digital image. Digital Image Forensics is a form of Image Analysis which deals with the problem of certifying the authenticity of a picture, or its origin. It can be roughly subdivided in to three branches:

- Image source identification, which aims to identify which device was used to capture an image (different models of scanner, digital camera, etc.);
- Computer generated image recognition, to detect if an image is natural or synthetic;
- Image tampering detection, to discover if an image has been intentionally modified by human intervention.

In this paper we focused on the problem of detecting duplicated regions in digital images. The region-duplication (or Copy-Move, see [1][2]]) is one of the most common forgery used for image tampering: a part of an image is copied and pasted into another part of the same image. This process is used to delete some objects from the scene, and to substitute information with some other taken from “good regions”, e.g. highly-textured areas or uniform ones. To make alterations harder to detect, post-processing techniques (i.e. smooth filters) are used, especially in the edges of the tampered areas. On the other hand, although these techniques may cause no visual

P. Foggia, C. Sansone, and M. Vento (Eds.): ICIAP 2009, LNCS 5716, pp. 893–901, 2009.

artifacts, they usually alters some image features. State of the art approaches proposed different features to be analyzed in order to detect these alterations.

Nevertheless the need of digital techniques for image authentication has been widely recognized, Digital Image Forensics is still a new research field in the Image Processing area. With regard to the image forgery detection problem, Farid proposed several statistical methods, based on color filter interpolation [3] and re-sampling [4]. Fridrich presented a solution to detect copy-move type of forgery [1]. Ng and Chang proposed a model for image splicing to detect photomontage [5].

To detect duplicate areas, the simplest approach is exhaustive search, but it is computationally expensive. To speed up the process, often digital images are not analyzed in the spatial domain, but projected in a different representation domain, and block-matching approaches are used. Some features are extracted from each block, and matched with those extracted by other blocks, in order to find those similar. In this paper we present a new method to find duplicated areas in digital images, by using bit-plane decomposition and block-matching analysis.

2 The Proposed Approach

In order to make the analysis process faster, images need to be represented in a convenient domain. We use bit-plane slicing to decompose images to analyze.

Our solution can be divided into three steps:

- Image decomposition by bit-plane slicing;
- Bit block encoding;
- Search for duplicated areas.

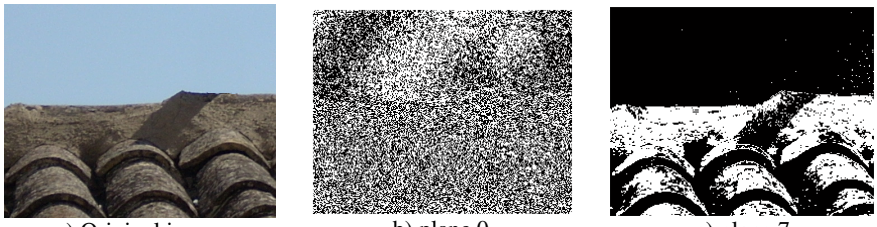
The next sub-sections will describe in detail each of these steps.

2.1 Bit Plane Decomposition

Bit-plane slicing is a well known technique used to represent the content of a grayscale image. It is mostly used for applications in the fields of digital watermarking [6] and image compression [7][8]. In [9] bit-planes are used to classify grayscale textures. In one of our previous works we used bit-plane representation for an image restoration application [10].

A n-bit grayscale image can be split in n different planes, one for each bit used to represent them. The higher bit-planes contain the most significant bits, that is the most part of the image information, while the lower ones usually contain noise (fig.1).

For our purposes, images to analyze are first grayscaled, therefore only the brightness component is processed. Image is then split in its bit-planes and Gray- coding is applied, to decorrelate information between different planes. We observed that working with block of bits in the bit-plane domain is simpler and faster than working with blocks of pixels of the image. Furthermore our method let the user to select the starting plane, to avoid processing less significant planes, speeding-up the execution time. In section 3 we will show how results are affected by setting the starting plane.



a) Original image b) plane 0 c) plane 7

Fig. 1. The most (c) and the less (b) significant bit-planes of an image (a)

2.2 Bit Block Encoding

Once the image has been decomposed in its bit planes, the next step is to use a suitable representation to simplify the following analysis step.

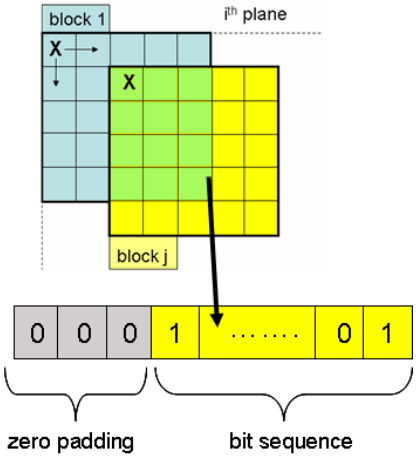


Fig. 2. Two overlapping blocks in i^{th}-plane. An m x m block is created from a starting point X (top part). Bit-blocks are rasterized and zero-padded (bottom).

We start from a user-selected plane and proceeds towards the most significant ones. The starting plane (size NxM) is analyzed in scan order (from top left to bottom right), and divided into m x m (m is also set by the user) overlapping blocks, each one shifted rightward by 1, and downward by 1 after processing a row. Image border bits are included only in the last blocks of each row. Therefore we have N' x M' blocks where N'=(N-m-1) and M'=(M-m-1).

Each block is a m x m binary matrix, and is rasterized in a array of m^2 bits. This array is zero-padded, in order to make its size multiple of 8 (fig.2). Bits are taken from the array 8 per time, and converted in a single char using ASCII code. Each block is coded into a string of $k=m^2/8$ chars (considering zero padding). Therefore the bit-plane is represented with a sequence of N' x M' strings with size k. Each string contains information about a block of bits. Note that there is a lot of redundancy, because blocks overlap, and closer ones differ only for a column or for a row. Bit-planes are analyzed in this coded domain, rather than bit per bit. We tested different types of coding (decimal, octal, hexadecimal), but ASCII code showed to be the most suitable for our goals. Bit-planes are represented in a very compact form, and larger blocks was used for our experiments. Starting plane is the only one to be entirely encoded. In the next section we will see that for the following planes only part of each plane is encoded, saving a lot of execution time (see section 2.3).

2.3 Searching for Duplicated Areas

With the previous steps, the starting plane is represented as a sequence of encoded strings. We sort this sequence, lexicographically, in order to have identical strings side by side. Then we analyze the whole sorted sequence, searching for groups of consecutive identical strings, and we mark these groups as possible candidate areas. The coordinates of the candidates are then passed up to the next plane, and bit-block encoding is applied only for bits in these candidate areas (see fig. 3), while the rest of the plane is simply ignored.

The process of encoding – searching candidates is repeated until the most significant plane is processed. The output of the last plane processing indicates the duplicated areas of the input image.

Choosing a starting plane different from the first one (plane 0) we can set the tolerance about the similarity measure used to search duplicated areas. If the starting plane is zero, detected areas are identical in the original grayscale image.

Note that the probability to have identical areas in less significant planes, which are typically similar to noise, is much lower than in most significant ones, except in case of tampering. Therefore starting from lower planes to the higher, rather than from higher to lower, reduce the number of candidates to be processed for the following planes, speeding up the process.

3 Experimental Results

We tested our method on a set of about 20 tampered color images, different both in their size and tampered area size.

For each test image, two types of test are executed:

- fixing the starting plane and varying block size;
- fixing block size and varying starting plane.

For each test we measured both accuracy and execution time.

As regards the accuracy, we created our image dataset copying and pasting parts of an image onto other parts of the same image. We saved source and destination area positions in a binary mask (see fig. 4.c, 5.c, 6.c), which represents our reference area A_R. Best results are those in which the detected area A_D is most similar to A_R. In particular we measured the detection precision *DP* as follows:

$$A_{In} = \frac{n(A_D \cap A_R)}{n(A_R)} \tag{1}$$

$$A_{Out} = \frac{n(A_D \cap \overline{A}_R)}{n(A_D)} \tag{2}$$

$$DP = A_{In} \cdot (1 - A_{Out}) \tag{3}$$

where:

- A_{In} is the ratio between the number of pixels in the intersection of the detected area A_D and the reference area A_R, and the number of pixels in A_R. When it tends to 1,

A_D covers the whole A_R, but nothing can be said about pixels outside A_R; if it tends to 0 A_D and A_R have smaller intersection;

- A_{Out} is the ratio between the number of pixels in A_D, which are not in A_R, and the number of pixels in A_D. When this parameter tends to 1, the whole detected area has no intersection with the reference. If it tends to 0, fewer pixels of A_D are labeled outside A_R. Nevertheless this parameter will not assure that the whole reference area has been covered.
- *DP* combines these two parameters: *DP* is high if A_D both covers A_R and has few outliers, and it is low if A_D and A_R are only partially overlapped or, though when A_R is well covered, A_D contains many pixels which are not in A_R.

Fig. 3 shows the effects of varying starting bit-plane (fig. 3.a) or block size (fig. 3.b), in detection precision. In the first case, if block size is fixed, we observed that the lower the starting plane, the higher the accuracy of our method. The drawback is an higher execution time because more planes have to be processed. In fact, the higher the starting plane, the looser the tolerance in searching similar areas, and therefore the larger the number of false positives (zones labeled as identical which are not identical) ($A_{Out} \rightarrow 1$). The "cut-off" starting plane P, that is the higher starting plane which gives best accuracy, depends on image resolution and typically is in [2,4].

If we vary block size, fixing the starting plane, two overlapping effects are observed:

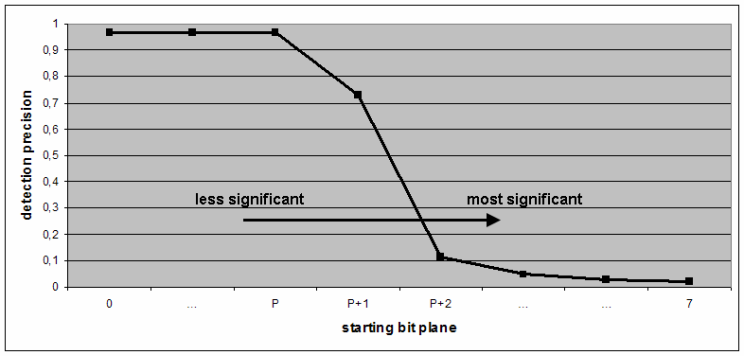
- If the block size is larger than the tampered areas, $A_{In} \rightarrow 0$. In fact, whatever block we consider in a plane, it will contain bits which do not belong to the tampered area. Therefore no matches can be found.
- If block size is small $A_{Out} \rightarrow 1$, because the probability to have natural similarities in an image, so that the number of false positives, increases.

The "steady state", [B'',B'], with the best accuracy, depends both on image resolution, and tampered area size and shape. B' is the largest block size to achieve best accuracy, B'' is the smallest one. For high-resolution images and smooth tampered area shape B' is larger than 16 and B'' 7-8. For low-resolution image with high-detailed tampered area shapes, B' is typically 6-7 and B'' 4, so that the "steady state" is very tight. In our experiments we measured an average accuracy in the "steady state" of 98,7%.

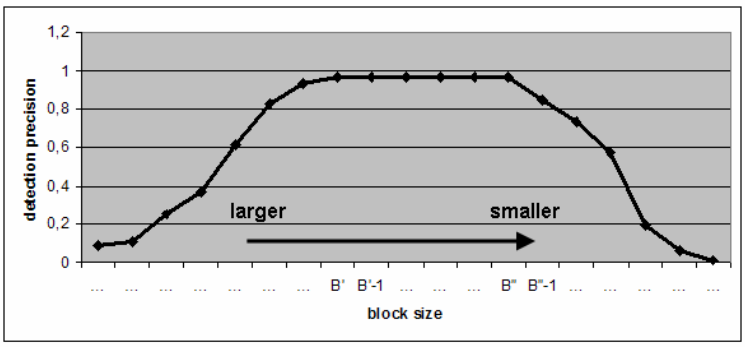
Fig.4 shows some detection results. Note that processing only the starting plane (fig.4.d), many false positives are detected. After processing some more planes, false positives are fewer. Final results is very similar to the reference area mask. For this image we measured an accuracy of 99,6%.

Fig.5 shows results obtained varying starting plane. Best results are achieved starting from plane 2 or lower (fig. 5. d, accuracy 97%). The higher the starting plane, the larger the number of false positives (fig. 5. e-i).

Fig.6 shows results obtained varying block size. Note that using larger block size (fig. 6.d,e,f) only part of the tampered area is detected. Best results with block size 8 (fig. 6.g), with an accuracy of 99,3%. Using smaller blocks (fig. 6.h,i), causes the detection of false positives.

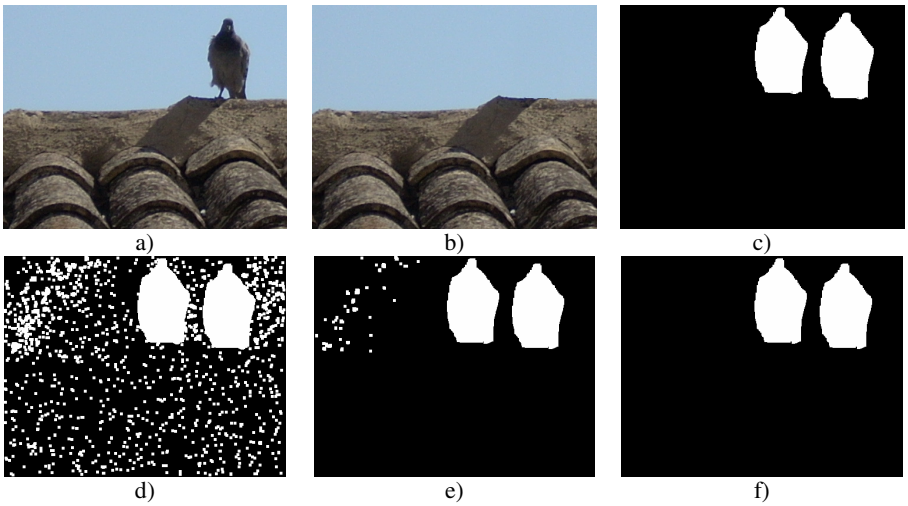


a)



b)

Fig. 3. Detection precision vs starting bit-plane (a) and vs block size (b). P is the "cut off" starting plane and depends on image resolution. B' and B'' are the two "cut off" block sizes, and depend on image resolution and on tampered area size.



a) b) c)

d) e) f)

Fig. 4. Detection results. Original image (a), tampered image (b), reference area mask (c), detected duplicated areas after processing the starting plane (d), intermediate result (e), final result (f).

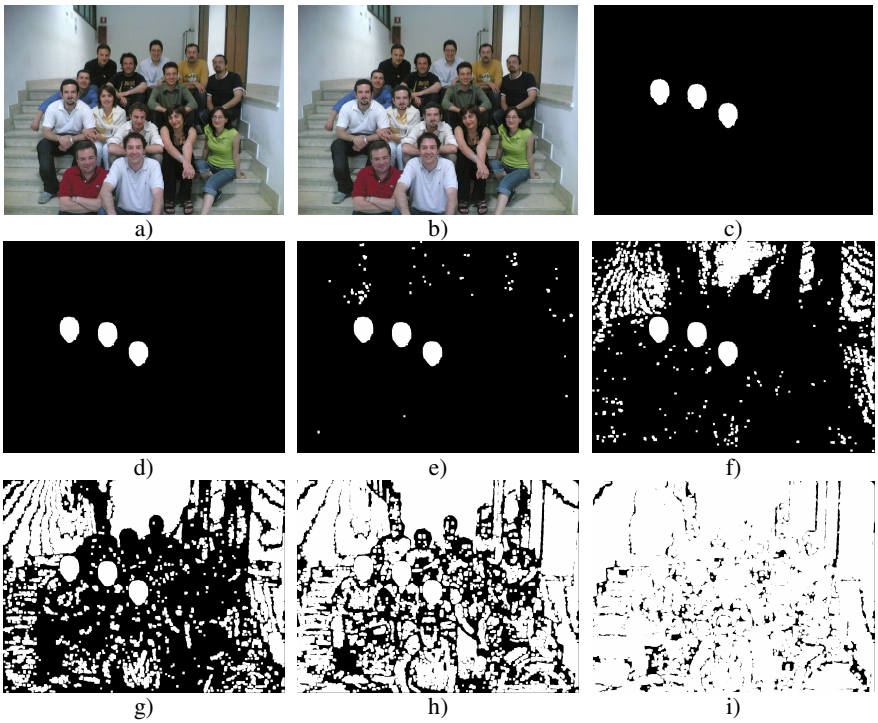


Fig. 5. Results varying starting plane. Original image (a), tampered (b), reference area mask (c), results with starting plane 2 (d), 3 (e), 4 (f), 5 (g), 6 (h), 7 (i).

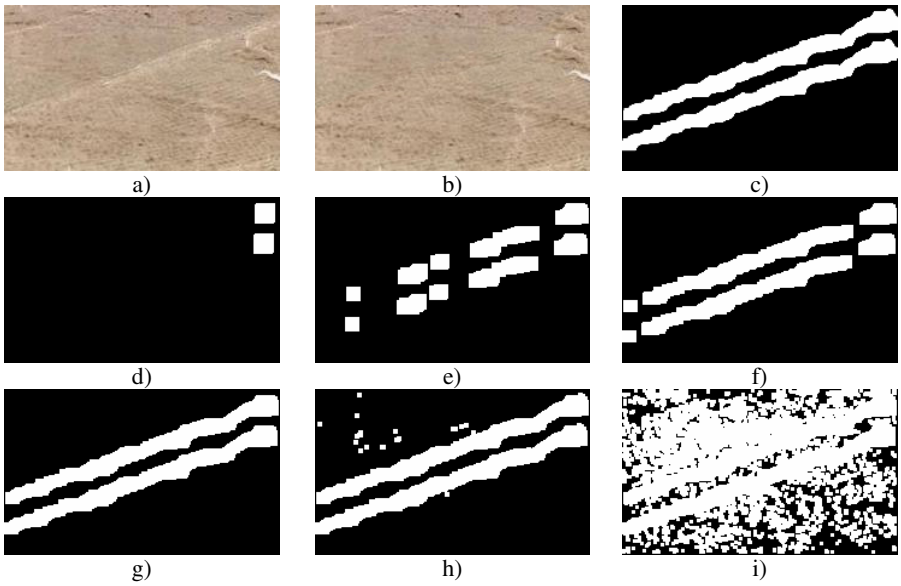


Fig. 6. Results varying block size. Original image (a), tampered (b), reference area mask (c), results with block size 16 (d), 14 (e), 10 (f), 8 (g), 4 (h), 3 (i).

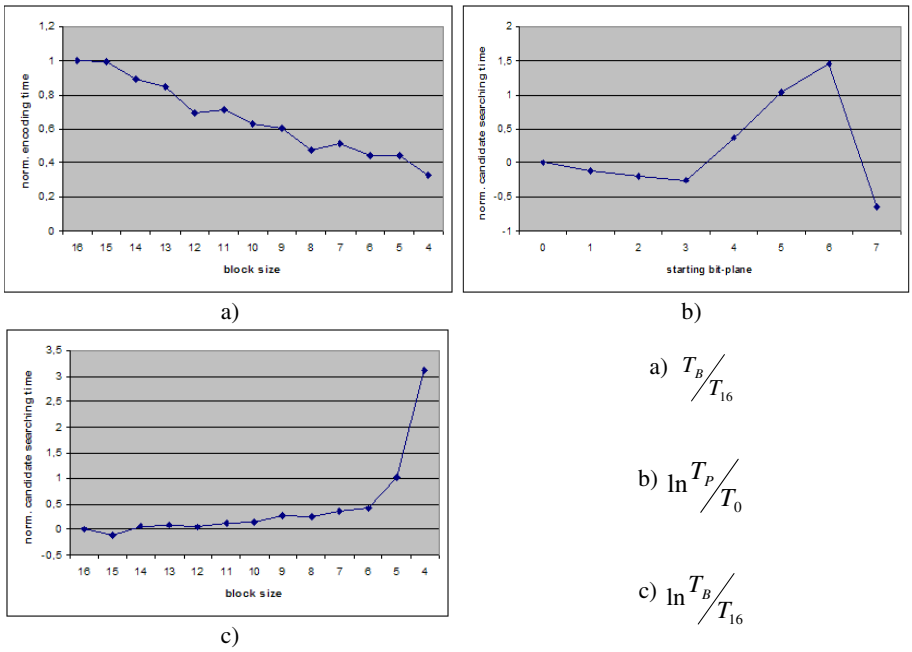


Fig. 7. (a) Execution time to encode the starting plane, varying the block size. (normalized with respect to the value measured for block size 16). (b) Candidate searching time vs starting plane. T_P is the time measured starting from bit plane P and T_0 is that for plane 0. (c) Candidate searching time vs block size. T_B is the time measured using block size B and T_{16} is that using block size 16. (we use for testing values of B between 4 and 16).

With respect to execution time, we separately measured time spent for each of the three parts of the algorithm: bit-plane decomposition, bit-block encoding (for the starting plane), and the candidate area searching. Decomposition time depends only on image size and takes about 0,5 s for small images (around 250x200) and few seconds for larger ones (around 1200x1000). Encoding time for the starting plane depends on image and block sizes (fig. 7.a), and it is independent from tampered area size. For the "steady state" it takes few seconds for small images up to few minutes for larger ones. We observed that whenever the square of the block size is multiple of 8 (e.g. 12, 8, 4) encoding time slightly decrease, because no zero padding step is needed. Candidate area searching time depends on both starting plane (fig.7.b), and block size (fig.7.c), in addition to image resolution and tampered area size. Starting analysis from higher planes reduces execution time, because fewer planes have to be analyzed. Nevertheless, when the starting plane is too high, false positives are detected and a larger area has to be analyzed in the following planes, spending more time. Starting from the most significant (plane 7) means searching duplicates only for that plane, so that time decreases again. Decreasing block size, execution time increases, because a larger number of duplicated areas is found. When block size is too small, many false positives are detected, and execution time rapidly increases. For the "steady state", candidate searching phase takes few seconds for small images up to half a minute for larger ones.

4 Conclusions and Future Works

Copy-move is one of the most used form of tampering to alter a digital image. Highly textured areas, or homogeneous ones, are typically used to delete objects from a scene or to replicate an object into the image.

In this paper we presented a new method to detect duplicated areas in a digital image. Our method analyzes a digital image in the bit-plane domain. Block of bits are encoded, using the ASCII code, into strings of characters, in order to find identical sequences in each bit plane. Detected candidate areas in a plane are processed in the following planes. Output of the last processed plane indicates tampered areas.

Experimental results showed that the proposed solution proves to reach very high accuracy without spending much execution time.

There are some drawbacks in our approach. First, it does not work with JPEG images, because JPEG compression alters (not uniformly) the intensity value of the pixels in an image. To our knowledge, there are no works about the relationship between JPEG compression and bit-plane representation. This is an interesting open problem. Moreover our method cannot be applied if the duplicated area is rotated or scaled. Solutions to detect other types of tampering will be the focus of our future studies. Furthermore our approach can be easily adapted to several more application fields: image compression, digital watermarking, image segmentation, etc.

References

1. Fridrich, J., Soukal, D., Lukas, J.: Detection of Copy-Move Forgery in Digital Images. In: Proc. Digital Forensic Research Workshop, Cleveland, OH (August 2003)
2. Mahdian, B., Saic, S.: Detection of Copy–Move Forgery Using a Method Based on Blur Moment Invariants, Forensic Science International, vol. 171(2), pp. 180–189
3. Popescu, A., Farid, H.: Exposing digital forgeries in color filter array interpolated images. IEEE Transactions on Signal Processing 53(10), 3948–3959 (2005)
4. Popescu, A.C., Farid, H.: Exposing Digital Forgeries By Detecting Traces of Re-Sampling. IEEE Transactions on Signal Processing 53(2), 758–767 (2005)
5. Ng, T.T., Chang, S.F., Sun, Q.: Blind Detection of Photomontage Using Higher Order Statistics. In: IEEE International Symposium on Circuits and Systems, Vancouver, Canada (May 2004)
6. Wang, R.Z., Lin, C.F., Lin, J.C.: Image hiding by optimal LSB substitution and genetic algorithm. Pattern Recognition 34, 671–683 (2001)
7. Kamata, S., Eason, R.O., Kawaguchi, E.: Depth-First Coding For Multivalued Pictures Using Bit-Plane Decomposition. IEEE Transactions on Communications 43(5), 1961–1969 (1995)
8. Galatsanos, N.P., Yu, S.S.: Binary Decompositions For High-Order Entropy Coding of Grayscale Images. IEEE Transactions on Circuits And Systems For Video Technology 6, 21–31 (1996)
9. García-Sevilla, P., Petrou, M., Kamata, S.: The use of Boolean model for texture analysis of grey images. Computer Vision and Image Understanding 74, 227–235 (1999)
10. Ardizzone, E., Dindo, H., Mazzola, G.: Texture Synthesis for Digital Restoration in the Bit-Plane Representation. In: The Third IEEE International Conference on Signal-Image Technology & Internet–Based Systems 2007, Shangai, China, December 16-19 (2007)